PATENT

TITLE: WINDOW MANAGER USER INTERFACE

Inventors: David SENECHALLE
Brian NOYES

Peter S.Weissman, Esq.
BLANK ROME COMISKY & MCCAULEY
900 17th Street, N.W., Suite 1000
Washington, D.C. 20006
(202) 530-7400

WINDOW MANAGER USER INTERFACE

BACKGROUND OF THE INVENTION

Field of the Invention

[0001] The present invention relates to a user interface and method for managing windows. More particularly, the present invention relates to a window manager used to simultaneously run and display multiple views of a single application or multiple applications.

Description of the Related Art

[0002] When users are monitoring or controlling multiple real time processes within a single application, each process is typically positioned within a separate window. In addition, when a user is operating multiple applications, each of those applications is positioned within a separate window. A problem therefore arises of how to best present the various windows involved.

[0003] Traditional user interfaces confine the user to essentially one window at a time. The user must then switch between the various windows by, for instance, selecting that window from the taskbar, or by minimizing the focus window and maximizing or restoring the desired window.

[0004] One type of window management system is referred to as Multiple Document Interface (MDI). MDI is used for window management within Microsoft Windows applications. Multiple child windows can be opened within a parent frame window. The child windows are overlapping windows within the bounds of the frame window, and can be

resized and rearranged. However, the user must arrange the child windows so that the bounds of the parent window are used efficiently.

[0005] MDI allows child windows to be maximized within the frame window, but only the current active or focus window can be maximized within the frame window and the remaining non-focused windows are hidden. To be seen, a hidden window must be activated or focused, which brings it to the front and hides the previously-focused window and other non-focused windows. Consequently, MDI requires the user to conduct a great deal of resizing and rearranging to achieve a layout of multiple windows.

[0006] Another type of window management system is referred to as a fixed split application. This system is used to arrange different views of data and controls within a parent window. However, these applications are limited to a fixed number of predefined splits that can occur. For instance, in Microsoft Outlook or Windows Explorer, the user can select a folder view to be displayed in a split window on the left. The content of whatever is selected in the left window is displayed in the right window. Other applications, such as Rational Rose or Borland JBuilder, present more than one split. However, the number, layout and selection of those splits is predefined, and the user is limited to just resizing the existing splits, not creating new ones.

[0007] A third type of window management system is referred to as the grid layout window. Many applications allow the user to automatically arrange open windows using a command known as "tile." The tile command lays out the open windows in a grid, tiling them either horizontally, vertically, or in a grid. The grid layout, however, enforces a rectangular grid of windows, and does not permit sub-splitting outside of that layout. The user cannot

resize a single pane, but can only alter the width of an entire row or column of similarly aligned panes. In addition, the user must re-initiate the tile command as new windows are created or removed.

[0008] Thus, these conventional user interfaces are either static or at best have a limited ability to create a separate window that overlaps with the first window. The prior systems require excessive user manipulations in order to manually layout the windows to maximize the use of the display screen. In addition, those prior art systems limit the number, layout and selection of windows that can be divided. Consequently, prior methods are clumsy, inflexible, and do not allow the user to easily view more than one window at a time.

SUMMARY OF THE INVENTION

[0009] In view of the foregoing, one object of the present invention is to provide a user interface for managing windows that is easy to use. Another object of the invention is to provide a user interface for managing windows that is flexible, and allows the user to simultaneously view multiple windows. Yet another object of the invention is to provide a window management system that does not limit the number of splits the user creates, the nesting scheme of those splits, or the resizing of the splits.

[0010] Still another object of the invention is to provide a window management system that automatically and dynamically arranges pane windows without user interaction, especially during creation and removal of pane windows. Another object of the invention is to provide a GUI framework for managing multiple real time processes from within a single or multiple frame windows without requiring the user to constantly switch between frame windows.

[0011] The window manager generally has a frame window, with one or more pane windows. Each pane window has a viewing area that is used to display an application. All of the pane windows are simultaneously presented to the user within the frame window. The pane windows do not interfere with one another, and are arranged to maximize the utilization of the space available in that frame window. If the application running in the viewing area cannot be fully viewed, a scroll bar and scroll keys are provided to enable the user to scroll through the entire viewing area.

[0012] The user can quickly and easily create and remove pane windows within the frame window. Pane windows are created by splitting an existing pane window either vertically or

horizontally, with the two new split pane windows occupying the space of the original pane window. When this is done, the original view occupies one of these split panes, and the other pane becomes available for an additional view. When a pane window is closed, one or more of the remaining pane windows are maximized to occupy its space. Accordingly, the window manager enables the pane windows to be dynamically and simultaneously presented to a user in a manner that maximizes the utilization of the space available in the frame window. As pane windows are split and closed, the window manager dynamically maximizes the resulting arrangement.

[0013] These and other objects of the invention, as well as many of the intended advantages thereof, will become more readily apparent when reference is made to the following description, taken in conjunction with the accompanying drawings.

[0014] BRIEF DESCRIPTION OF THE DRAWINGS

[0015] Figure 1 shows a frame window having multiple pane windows in accordance with the preferred embodiment of the invention.

[0016] Figure 2 is a flow chart depicting the overall operation of the window manager.

[0017] Figure 3 is a flow chart showing the operation for inserting a pane window into the frame window.

[0018] Figure 4 is a flow chart showing the operation for splitting a pane window.

[0019] Figure 5 is a flow chart showing the operation for closing a pane window.

[0020] Figure 6 is a flow chart showing the operation for floating a pane window.

[0021] Figure 7 is a flow chart showing the operation for swapping views between two pane windows.

[0022] Figure 8 is a flow chart showing the operation for moving a view into a new split pane window.

[0023] Figure 9 is a flow chart showing the operation for resizing a pane window.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0024] In describing a preferred embodiment of the invention illustrated in the drawings, specific terminology will be resorted to for the sake of clarity. However, the invention is not intended to be limited to the specific terms so selected.

[0025] Turning to the drawings, an overview of the window manager 10 is shown in Fig. 1. The window manager 10 generally has a frame window 12, with one or more pane windows 14. Each pane window 14 has a viewing area 16 that is used to display an application. All of the pane windows 14 are simultaneously presented to the user within the frame window 12. The pane windows 14 do not interfere with one another, and are arranged to maximize the utilization of the space available in that frame window 12. If the application running in the viewing area 16 cannot be fully viewed, a scroll bar and scroll keys are provided to enable the user to scroll through the entire application.

[0026] The pane windows 14 are separated by one or more posts 18 that define boundaries between the pane windows 14. The posts are not visible to the user, and are only included in the figure to best illustrate and describe the invention.

[0027] Each pane window 14 is also provided with accelerator keys or command icons 22, 24 and 26. The command icons 22, 24, 26, permit the user to control the number and position of the frame windows 12 and pane windows 14. The horizontal split command 22 divides the pane window 14 into two horizontally-aligned pane windows 14. The vertical split command 24 divides the pane window 14 into two vertically-aligned pane windows 14. The division is preferably into two equal parts, though any suitable division can be made, such as into more than two pane windows, or into pane windows that are unequal in size.

[0028] The float command 26 removes the pane window 14 from the frame window 12. The view that was removed from the frame window 12 then appears as the only view of a new frame. The "floated" pane window 14 is then referred to as a secondary frame window 12 and the frame window 12 from which it was created is referred to as the primary frame window 12. The secondary frame window 12 is no longer within the confines of the primary frame window 12, and therefore the primary and secondary frame windows can overlap one another. The secondary frame window 12 can be further divided into one or more pane windows 14 (divisible mode), or can be designed as being non-divisible (non-divisible mode).

[0029] Under the divisible mode of operation, the secondary frame window 12 has the life of the primary frame window 12, so that if the primary frame window 12 is closed, then the secondary frame window 12 is automatically closed. However, the life of the primary frame window 12 is independent of the life of the secondary frame window 12, so that if the secondary frame window 12 is closed, the primary frame window 12 remains focused. Under the non-divisible mode of operation, either frame window 12 can be closed without closing the other frame window 12. If the last remaining frame window 12 is closed, the application ends.

[0030] The embodiment of Fig. 1 includes one frame window 12 and three pane windows 14, which are labeled as Window A, Window B and Window C. When the frame window 12 is initially opened, the pane window, Window A, encompasses the entire frame window 12. At that time the user selected to split Window A horizontally. As a result two horizontally-aligned pane windows were created, and separated by the long horizontal post 18 that extends

the entire width of the frame window 12. Next, the lower pane window was split vertically, creating Window B and Window C, which are separated by the vertical post 18.

[0031] After each pane window 14 was created, the user opened an application in the view 16 of that pane window 14. For simplicity, each of the pane windows 14 is shown having the same application. The buttons Restart, Stop, Purge, Scan and Restore relate to the operation of that application. However, the pane windows 14 can each have different applications that operate simultaneously.

[0032] The frame window 12 includes a toolbar that enables the user to perform standard operations, such as to start an application, close an application, or print, in the focused pane window. The user can also save the frame window 12 configuration, including the arrangement of pane windows 14, and the current application parameters.

[0033] The operation of the system will now be discussed with reference to Fig. 2, where the overall operation of the system is shown at 200. The user starts the application and opens a frame window 12. At this point, the user can either open a pane window 14 or a configuration that was previously saved by the user. If the user chooses to insert a pane window 300, operation continues at step 302 of Fig. 3. At step 302, the window manager 10 determines whether the frame window 12 is empty. If the frame window 12 is not empty, such as when the user opens a saved configuration, the pane window 14 is not inserted into the frame window 12, step 304, and operation returns to point A of Fig. 2.

[0034] On the other hand, if the frame window 12 is empty at step 302, then the new pane window is created, step 306, and inserted into the frame window 12, step 308. The newly-created pane window 14 is sized to fill the frame window 12, step 310. Operation then returns

to point A of Fig. 2. In an alternative embodiment, the window manager 10 can automatically open a pane window 14 when the frame window 12 is opened. If the user wishes to use a saved configuration, that configuration is restored.

[0035] Once the pane window 14 has been opened, either by inserting the pane window 300, opening a saved configuration, or being automatically opened by the window manager 10, the user has the following options: split a pane window 400, close a pane window 500, float a pane window 600, swap views between two pane windows 700, move view into new split pane window 800, or resize a frame window or pane window 900. These options continue until the last pane window 14 or the frame window 12 is closed.

[0036] If the user chooses to split a pane window 14, step 400, processing continues in Fig. 4. The user may choose to split a pane window 14 by selecting the horizontal split command or the vertical split command from the pull-down program menu or via the shortcut command icons 22, 24, respectively. At step 402, the system then analyzes the pane window 14 that is to be split to determine whether the split would violate the minimum size of that pane window 14. The window manager 10 automatically sets the minimum size for each pane window 14 when the pane window 14 is created. However, that preset size can be re-programmed by the user. The minimum size ensures that the application running in that pane window 14 will have a sufficient size to be viewable to the user. Accordingly, if the split would result in the pane window 14 having a size less than a preset or predetermined size for that window, the split is disallowed, step 404, and operation returns to point A of Fig. 2.

[0037] Assuming the minimum size for the pane window 14 to be split would not be violated, step 402, the window manager 10 then splits that pane window into two pane

windows, step 406. The split is made horizontally or vertically, depending on whether the user chose the horizontal or vertical split command 22, 24. The split creates two pane windows 14 in the space that was previously occupied by the single pane window 14.

[0038] The window manager 10 then determines whether the minimum size of one of the views is larger than one-half of the original space of the original pane window, step 408. This would occur, for instance, if the original view has a smaller minimum size than the view inserted by default into a new split pane window. As long as the sum of the minimum widths and heights of the two split pane windows is less than the width and height of the original pane window, the split is allowed to occur. If the minimum width and height for one of the split windows exceeds one-half of the respective width and height of the original pane window, the widths and heights for the split pane windows are set at the larger of the two minimum widths and heights, step 412. Otherwise, the split is set at one-half the size of the original pane window, step 410. However, if the minimum width and height for each of the split views exceeds one-half of that of the original pane window, the split is disallowed.

[0039] Once the original pane window is split into the two pane windows 14, the original view is placed into one of the new split pane windows, step 414. At step 416, the other pane window 14 can have a blank view, or contain a selector view that enables the user to open an application into that pane window 14. The window manager 10 can be set to define which pane window will contain the original view, and the user can move the view into the empty pane window. The system operation then returns to point A in Fig. 2.

[0040] If the user at any time decides to close a pane window 14, operation proceeds at step 500 of Fig. 5. The user can close a pane window 14 by selecting the close command

from the pull-down program menu or via a shortcut command icon located within that pane window 14. Once the user selects to close a pane window 14, that pane window is removed from the frame window 12, step 502. The remaining pane windows 14 are then expanded to fill the space that was previously occupied by that closed pane window, step 504, and the closed window view is disposed or returned to a framework that resides at a higher level than the window manager, step 506. The operation of the system then returns to point A in Fig. 2. However, if the pane window was the only pane window in the frame window 12, then the window manager 10 waits for a new pane window to be inserted 300, Fig. 2. Alternatively, the window manager 10 can prevent the last pane window 14 from being closed.

[0041] Accordingly, the window manager 10 maximizes the space within a frame window 12. Preferably, the window manager 10 expands one or more neighboring pane windows 14 that have the same height and/or width. For instance, if the user closes Window B of Fig. 1, Window C would be expanded to the space occupied by Window B since Window C has the same height as Window B. Window A would remain unchanged. If, on the other hand, Window A were closed, then both Windows B and C could be expanded vertically to occupy the space of Window A.

[0042] If the user chooses to float a pane window 14, step 600, processing continues in Fig. 6. The user may choose to float a pane window 14 by selecting the float command from the pull-down program menu or via the shortcut float command icon 26. When a user decides to float a pane window 14, the pane window 14 is removed from the current frame window 12, step 602, in the same manner as for closing a pane window, namely steps 502-506 (Fig. 5). In addition, a secondary frame window 12 is created, step 604, and the pane window 14 is

inserted into the secondary frame window 12, step 606. Thus, the pane window 14 is removed from the primary frame window 12, and inserted into the new secondary frame window 12.

[0043] Since no other pane windows are in the secondary frame window 12, the pane window 14 that was floated is maximized to encompass the entire frame window 12. The window manager 10 can be programmed so that the secondary frame window 12 is either non-divisible, or that it can be divisible into further pane windows 14 and/or floated frame windows 12. If the secondary frame window 12 is divisible, step 608, the respective split and/or float commands are made available in the secondary frame window 12, step 610. Otherwise, the commands are not made available in the secondary frame window, step 612. Operation returns to point A of Fig. 2.

[0044] The user also has the option of switching views 16 between two pane windows 14, step 700, as detailed in Fig. 7. In other words, the user can switch the applications running in two different pane windows 14 by moving the applications to the other pane window 14. The two pane windows can be in the same frame window or in different frame windows. To do this, the user presses and holds the mouse button in the client area (that is, the application) of one pane window, step 702. The mouse cursor, and thus the application, is dragged into the target pane window 14, step 704, and released into the target pane window 14, step 706. The window manager 10 automatically switches the applications between the source pane window and the target pane window, steps 708, 710. The operation continues at point A of Fig. 2.

[0045] The user also has the option of moving a view 16 (that is, application or program) into a new split pane window 14 that may be in either the same frame window or a different

frame window, step 800, Fig. 8. Here, the user is not only moving the view 16, but also closing the source pane window and creating a new split pane window at the same time. To do so, the user selects the source view, such as by pressing the mouse button on the view to be moved while holding down the CTRL key, step 802.

[0046] The mouse cursor, and thus the view, is dragged to another pane window 14, step 804, and released when the mouse button is released, step 806. Each pane window 14 is divided into four quadrants, as shown by the dashed X in Window B of Fig. 1. The pane window 14 is split in the direction identified by the quadrant in which the view was released, step 808. Thus, for instance, if the source view was released in the upper quadrant, the pane window 14 is split vertically. However, if the target pane window is in the non-divisible mode of operation, then the source pane window is deleted from the source frame window and the target pane window is replaced by the source pane window, and no split occurs. In addition, if the user releases the view into an empty pane window 14, there is no need to split that pane window 14, and the view is simply moved into that empty pane window.

[0047] After the pane window 14 is split, step 808, the source pane window 14 is removed from the parent frame window 12, step 810. The source pane window 14 is removed in the same manner as for closing a pane window, at steps 502-506 of Fig. 5. The view from the source pane window 14 is inserted into the new split pane window 14, which occupies the location of the drop quadrant, step 812. The view that occupied the target pane window is placed into the split pane window 14 that is opposite the drop quadrant, step 814. Operation proceeds at point A of Fig. 2.

[0048] At any time, the user can resize a pane window 14 or frame window 12, step 900, Fig. 9. To do so, the user places the mouse cursor over the border between the pane windows 14 to be resized, or at the edge of the frame window 12 to be resized, step 902. A resizing guide will appear, and the user presses and holds the mouse button, step 904, and drags the border to the desired position, step 906. If the user is resizing the width of the pane window or frame window, the vertical border is dragged, but can only be dragged horizontally. If the user is resizing the height of the pane window or frame window, the horizontal border is dragged in a vertical direction. The user can also place the mouse cursor on a corner, and the resizing guide allows the user to simultaneously resize the horizontal and vertical dimensions.

[0049] If the user is resizing a pane window 14, the window manager 10 determines whether the minimum size constraint of any pane window would be violated, step 908. If the resizing would violate the minimum size constraint of a pane window, the resizing is terminated at the current position, step 910, and further resizing is disallowed in that direction, step 912. If the minimum size constraint is not violated, step 908, then the user eventually releases the mouse button, step 914, and the resizing is terminated at that position, step 916. Operation proceeds at point A of Fig. 2.

[0050] If the user is resizing a frame window 12, the window manager resizes the pane windows 14 along the bottom and right borders of the frame window 12 as the frame window 12 is resized. Only the pane windows at the right edge or bottom edge of the frame window are resized. Once these pane windows reach its minimum size in a particular dimension, step 908, the frame window cannot be made smaller in that dimension, step 910, until the user readjusts the size of that pane window. If the minimum size of the pane windows is not

violated, the resizing is stopped at the position of the cursor when the user releases the mouse button.

[0051] Although resizing is preferably implemented by use of mouse control, resizing can also occur through the use of keyboard commands. The user can place the window manager 10 into a resizing mode through the use of a command key or key combination (such as CTRL-R). Once in the resizing mode, the arrow keys are used to first select which direction the window is being resized in, and then to resize the pane window 14. Another key, such as the ESC key, can complete the resizing operation. Keyboard commands can also be used to define which pane window is presently focused (such as CTRL-TAB to scroll through the various pane windows). Different keys can be used to control sizing of the frame window, as opposed to the pane windows.

[0052] Keyboard controls can also be used for the other functions, such as inserting a pane window 300, splitting a pane window 400, closing a pane window 500, floating a pane window 600, swapping views between two pane windows 700 and moving a view into a new split pane window 800. For instance, for swapping views and moving and splitting panes, a key combination is used to initiate the drag and drop operations, then the arrow keys are used to position the cursor, and another key combination completes the drag and drop operation.

[0053] Accordingly, the window manager enables the pane windows 14 to be dynamically and simultaneously presented to a user in a manner that maximizes the space available in the frame window 12. As pane windows 14 are split and closed, the window manager dynamically maximizes the resulting arrangement.

[0054] Though the applications running in each pane window 14 are operating simultaneously, the user can generally only work in one pane window 14 at a time. The pane window 14 that is focused is depicted by changing the color of the caption bar at the top of that pane window 14. For instance, the color of the caption bar is blue when focus resides in that pane window, and dark gray if the pane window last had the focus (such as during a menu selection or if another frame or application gets the focus) and light gray if the pane window does not have the focus. In the embodiment of Fig. 1, Window C is shown as having the focus, and Windows A and B are non-focused.

[0055] Any command chosen from the pull-down program menu is enacted on the pane window 14 having the focus. The user can select which pane window 14 is to be focused by clicking a mouse in that pane window 14, by selecting the pane window 14 from the pull-down program menu, or by using a key command that scrolls through the various pane windows 14.

[0056] At any time, the user can close the entire application, including all frame windows. At that time, the window manager saves the configurations of all the frame windows, including the layout of pane windows 14 for the primary frame window 12 and any secondary frame windows 12, as well as the settings for the applications running in each pane window 14. The saved configuration is automatically restored for the user whenever the application restarts.

[0057] If, during any of the operations 400-1000, that operation is interrupted or canceled by the user, the operation is canceled and the operation continues at point A of Fig. 2 as if the operation had not been undertaken.

[0058] Though the invention preferably splits a pane window into two pane windows, additional splits can be made. Accordingly, the user can specify any number of splits for a single operation. The pull-down menu can allow the user to select the desired number, and clicking on and holding down the command icon 22, 24 can result in a pull-down of the number of desired splits. Thus, for instance, the user can split a single pane window 14 into three equally sized pane windows in a single operation.

[0059] Another feature of the present invention is to provide a view selector interface into an empty pane window. The view selector interface permits the user to select a view (such as an application), which then replaces the view selector interface. However, the view selector interface can also be presented in a different pane window 14 or in a different frame window 12. The view selector interface can be designed to respond to messages from other program elements, such as from menu selection or a message sent by an application outside the control of the window manager.

[0060] In addition, a command is alternatively provided that permits the user to maximize any one or more pane windows 14 to encompass the entire frame window 12. The maximized pane window 14 hides the other pane windows 14. When the user restores (un-maximizes) a pane window 14, the window manager returns that pane window to its original size and position. If a pane window is closed from the maximized state, the new layout of the remaining pane windows is determined as if the removed pane windows were individually closed in the normal split layout.

[0061] The present invention can be used to simultaneously display a single application in the pane windows, or different applications in one or more pane windows. Additionally, the

invention can be implemented within an application that otherwise does not have the window management features of the present invention, such as the ability to split pane windows. Toward this end, an Application Programming Interface (API) is provided that allows the user to quickly develop an application into a window manager. The application derives the classes from a specified set of classes that are implemented in the API.

[0062] The base classes provided by the API include an Application class and a Frame class. The user derives the application from the Application class and each frame from the Frame class. However, the view windows of the application are not restricted to being based on a class of the API. The user provides these views, whose function is to incorporate the information processing specific to the application. The application can provide whatever behavior and functionality is desired through the derived application, frame and view classes. The API classes themselves provide all the functionality of the window management.

[0063] The API provides the application with the functionality to handle window management, including inserting pane windows, splitting pane windows, removing pane windows, resizing pane windows and frame windows, floating pane windows into a new frame, swapping views between pane windows, and moving a pane window from one location into a new split in an existing pane window.

[0064] The API includes an application development wizard that allows the user to create a skeletal application by indicating certain information, such as application name, desired names for the derived classes, and the file names and locations for the created code files. At the completion of the application development wizard, code is created that is compiled and run to have an empty application that provides all the functionality of the window manager

10. The user then specifies the views that are to be inserted into the pane windows, and defines the events that occur in those views.

[0065] The foregoing descriptions and drawings should be considered as illustrative only of the principles of the invention. The invention may be configured in a variety of shapes and sizes and is not limited by the dimensions of the preferred embodiment. Numerous applications of the present invention will readily occur to those skilled in the art. Therefore, it is not desired to limit the invention to the specific examples disclosed or the exact construction and operation shown and described. Rather, all suitable modifications and equivalents may be resorted to, falling within the scope of the invention.